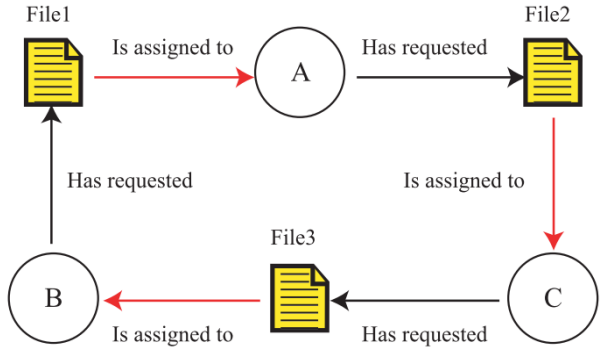
Introduction to Computer Science-103

Final exam

1. Three processes (A, B, and C) are running concurrently. Process A has acquired File1, but needs File 2. Process B has acquired File3, but needs File 1. Process C has acquired File2, but needs File3. Draw a diagram for these processes. Is this a deadlock situation? (6%) (P7-9)

   This is a deadlock situation (see Figure P7-9) because all four conditions of deadlock (mutual exclusion, resource holding, no preemption, and circular waiting) are all present.

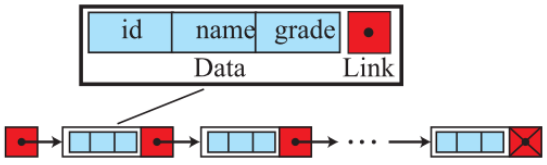
Figure P7-9 *A deadlock situation*



2. Draw a diagram to show a linked list in which the data part is a student record with three fields: *id*, *name*, and *grade*. (6%) (P11-13)

   The linked list of records

Figure P11-13 *Linked list of records*



3. Write an algorithm in pseudocode to compare the contents of two stacks. (8%) (P12-7) Checking the equality of two stacks

```
Purpose: Check if two stacks are the same
Pre: Given: S1 and S2
Post:
Return: true (S1 = S2) or false (S1 ≠ S2)
{
    flag ← true
    Stack (Temp1)
    Stack (Temp2)
    while (NOT empty (S1) and NOT empty (S2))
    {
        pop (S1, x)
        push (Temp1, x)
        pop (S2, y)
        push (Temp2, y)
        if (x ≠ y)
            flag ← false
    }
    if (NOT empty (S1) or NOT empty (S2))
        flag ← false
    while (NOT empty (Temp1) and NOT empty (Temp2))
    {
        pop (Temp1, x)
        push (S1, x)
        pop (Temp2, y)
        push (S2, y)
    }
    return flag
}
```

4. Figure.1 is a source code translation process. A source is compiled by means of four tools which are lexical analyzer, syntax analyzer, semantic analyzer, code generator, respectively. What is the corresponding tool of each phase? (4%) (Figure 9.1)

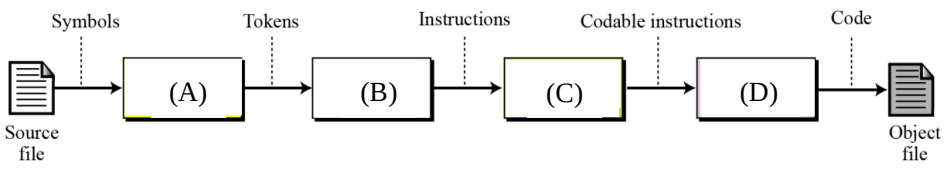
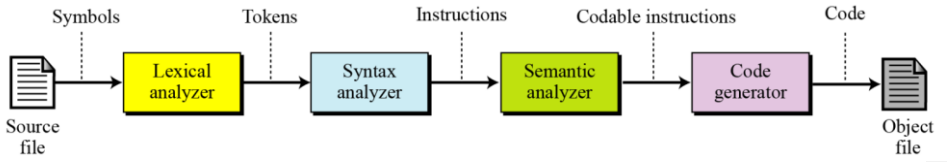


Figure .1



5. Please follow the program to show the results. (4%)

```
int x = 0;
if (x = 0 || x == 0)
        printf("%d\n", x);
printf("%d\n", x);
```

1

1

6. Write an algorithm in pseudocode to apply **binary search** on an array of elements. (6%) (P11-5)

The algorithm shows the binary search routine in pseudocode (see Chapter 8). Note that we perform the binary search on sorted array. If flag is true, it means x is found and i is its location. If flag is false, it means x is not found; i is the location where the target supposed to be.

```
Algorithm: BinarySearchArray (A, n, x)
Purpose: Apply a binary search on an array A of n elements
Pre: A, n, x                          // x is the target we are searching for
Post: None
Return: flag, i
{
    flag ← false
    first ← 1
    last ← n
    while (first ≤ last)
    {
        mid = (first + last) / 2
        if (x < A[mid])
            Last ← mid − 1
        if (x > A[mid])
            first ← mid + 1
        if (x = A[mid])
            first ← Last + 1                  // x is found
    }
    if (x > A[mid])
        i = mid + 1
    if (x ≤ A[mid])
        i = mid
    if (x = A[mid])
        flag ← true
    return (flag, i)
}
```

7. If the subprogram *calculate* (*A*, *B*, *P*, *S*) accepts the value of *A* and *B* and calculates their sum *S* and product *P*, which variable do you pass by value and which one by reference? (6%) (P9-23)

   A and B should be passed by value, S and P by reference.

8. Write an algorithm to delete an element in a sorted array. The algorithm must call a search algorithm to find the location of insertion. (6%) (P11-7)

   The algorithm that insert an element in a sorted array has two parts. Part a shows the main algorithm. Part b shows the algorithm named shiftup called by the insert algorithm.

   a. Algorithm P11-7a shows the main algorithm.

Algorithm P11-7a *Main algorithm for insertion*

```
Algorithm: DeleteSortedArray (A, n, x)
Purpose: Delete an element from a sorted array
Pre: A, n, x  // x is the value we want to delete
Post: None
Return:
{
    {flag, i} ← BinarySearch (A, n, x)      // Call binary search algorithm
    if (flag = false)                       // x is not in A
    {
        print (x is not in the array)
        return
    }
    ShiftUp (A, n, i)              // Call shift up algorithm
    return
}
```

   b. Algorithm11-7b shows the auxiliary algorithm used by the main algorithm.

   The shift-up algorithm used by the insert algorithm

```
Algorithm: ShiftUp (A, n, i)
Purpose: Shift up all elements one place up from index i.
Pre: A, n, i
Post: None
Return: A
{
    j ← i
    while (j ≤ n + 1)
    {
        A[j] ← A[j + 1]
        j ← j + 1
    }
    return
}
```

9. A mono-programming operating system runs programs that on average need 10 microseconds access to the CPU and 70 microseconds access to the I/O devices. What percentage of time is the CPU idle? (6%)

   70 / (70 + 10) × 100 = 87.5%

10. Using the selection sort algorithm, manually sort the following list and show your work in each pass using a table: (6%) (P8-5)

14	7	23	31	40	56	78	9	2

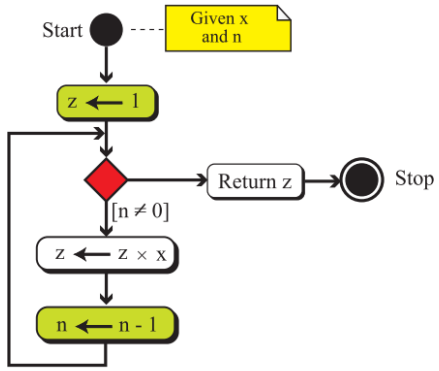
    The status of the list and the location of the wall after each pass of the selection sort algorithm is shown below:

Pass	List								
	14	7	23	31	40	56	78	9	2
1	***2***	7	23	31	40	56	78	9	14
2	***2***	***7***	23	31	40	56	78	9	14
3	***2***	***7***	***9***	31	40	56	78	23	14
4	***2***	***7***	***9***	***14***	40	56	78	23	31
5	***2***	***7***	***9***	***14***	***23***	56	78	40	31
6	***2***	***7***	***9***	***14***	***23***	***31***	78	40	56
7	***2***	***7***	***9***	***14***	***23***	***78***	***40***	78	56
8	***2***	***7***	***9***	***14***	***23***	***78***	***40***	***56***	78

11. Using the UML diagram for the product algorithm, draw a diagram to calculation the value of x^n, when x and n are two given integers. (6%) (P8-38)

    Figure 8-38 shows the UML for finding the power of an integer with an integral exponent.

Figure P8-38 *Power*



12. Please follow the program to show the results. (6%)

```
#include <stdio.h>
void delete_element (int value, int&array_size, int array[]){
	int i,;
	int location = array_size + 1;

	for(i = 0;i< array_size;i++){
		if (array[i]==value)
			location = i;
	}
	for (i = location ;i<= array_size;i++)
		array[i] = array[i+1];

	array_size--;
}
int main(){
	int i;
	int arraysize = 10;
	int a[10] = {0,1,2,3,4,5,6,7,8,9};

	for (i=0;i<arraysize;i++)
		printf("%d  ",a[i]);

	delete_element(5,arraysize,a);
	printf("\n") ;

	for (i=0;i<arraysize;i++)
		printf("%d  ",a[i]);

	return 0;
}
```

```
0  1  2  3  4  5  6  7  8  9
0  1  2  3  4  6  7  8  9
```

13. Please show how to build a linked list from scratch using the insertion algorithm (by following algorithm 11.3) (6%) (P11-16)

Algorithm 11.4 Inserting a node in a linked list

```
Algorithm: InsertLinkedList (list, target, new)
Purpose: Insert a node in the linked list after searching the list for the right
position
Pre: The linked list and the target data to be inserted
Post: None
Return: The new linked list
{
      searchlinkedlist (list, target, pre, cur, flag)
      // Given target and returning pre, cur, and flag

      if (flag = true)   return list              // No duplicate
      if (list != null                            // Insert into empty list
      {
            list ← new
      }

      if (pre = null)                             // Insertion at the beginning
      {
            (*new).link ← cur
            list ← new
            return list
      }

      if (cur = null)                             // Insertion at the end
      {
            (*pre).link ← new
            (*new).link ← null
            return list
      }

      (*new).link ← cur                           // Insertion in the middle
      (*pre).link ← new
      return list
}
```

Algorithm P11-16 shows the routine in pseudocode for building a linked. The routing uses the InsertLinkedList algorithm.

Algorithm P11-16

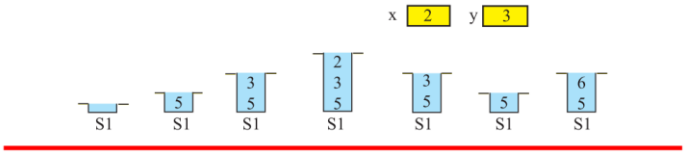
```
Algorithm: BuildLinkedList(data records)
Purpose: Build a linked list from scratch
Pre: given list of data records
Post: None
Return: list, which is a pointer pointing to the first node
{
    list ← null
    while (more data records)
    {
        InsertLinkedList (list, next record.key, next record)
    }
    return (list)
}
```

14. Show the contents of stack S1 and the value of variable x and y after the following algorithm segment is executed. (6%) (P12-5)

```
stack (S1)
Push (S1,5)
Push (S1,3)
Push (S1,2)
if (not empty (S1))
{
      pop(S1, x)
}
if (not empty (S1))
{
      pop(S1, y)
}
push (S1, 6)
```

Figure P12-5 shows the contents of the stack and the value of the variables.

Figure P12-5 *Solution to problem P12-5*



15. Write an algorithm to find the average of the numbers in a linked list of numbers. (6%) (P11-17)

Algorithm P11-17 shows the routine for finding the average of a linked list.

Algorithm P11-17

```
Algorithm: LinkedListAverage (list)
Purpose: Evaluate average of numbers in a linked list
Pre: list
Post: None
Return: Average value
{
    counter ← 1
    sum ← 0
    walker ← list
    while (walker ≠ null)
    {
        sum ← sum + (*walker).data
        walker ← (*walker).link
        counter ← counter + 1
    }
    average ← sum / counter
    return average
}
```

16. Write an algorithm that reverses the elements of an array so that the last element becomes the first, the second to the last becomes the second, and so forth. (6%) (P11-2)

Algorithm P11-2 shows the routine in pseudocode that reverses the elements of an array.

Algorithm P11-2 *Reversing elements of an array*

```
Algorithm: ReverseArray (A, n)
Purpose: Reverse the elements of an array
Pre: Arrays A with n elements
Post: Array A with elements reversed
Return:
{
    i ← 1
    j ← n
    while (i < j)
    {
        Temp ← A[j]
        A[j] ← A[i]
        A[i] ← Temp
        i ← i + 1
        j ← j – 1
    }
}
```

17. Change the following code segments to use an *if-else* statement: (6%)

註解 [m1]: 新增此題

```
#include < stdio.h >
int main()
{
	float a,b,ans;
	char key;
	printf("input two number:");
	scanf("%f %f",&a,&b);
	printf("press +,-,*,/:");
	key=getch();
	switch(key)
	{
	case '+':
		ans=a+b;
		break;
	case '-':
		ans=a-b;
		break;
	case '*':
		ans=a*b;
		break;
	case '/':
		ans=a/b;
		break;
	default:
		printf("Undefined key\n");
		exit(0);
	}
	printf("%f%c%f=%f\n",a,key,b,ans);
	return 0;
}
```

```
#include < stdio.h >
#include < conio.h >
void main(void)
{
        float a,b,ans;
        char key;
        printf("input two number:");
        scanf("%f %f",&a,&b);
        printf("press +,-,*,/:");
        key=getch();
        if ( key=='+' )
        {
                ans=a+b;
        }
        else if ( key=='-' )
        {
                ans=a-b;
        }
        else if ( key=='*' )
        {
                ans=a*b;
        }
        else if ( key=='/' )
        {
                ans=a/b;
        }
        else
        {
                printf("Undefined key\n");
                exit(0);
        }
        printf("%f%c%f=%f\n",a,key,b,ans);
}
```